ARMY RESEARCH LABORATORY

The Inertial Reticle Technology (IRT) Applied to a Remington 700 Sniper Rifle

by Timothy L. Brosseau, Mark D. Kregel,
Bailey T. Haug, and John T. McLaughlin

ARL-TR-2231

May 2000

Army Research Laboratory
Aberdeen Proving Ground, MD 21005-5066

ARL-TR-2231 **May 2000**

The Inertial Reticle Technology (IRT) Applied to a Remington 700 Sniper Rifle

Timothy L. Brosseau, Mark D. Kregel, Bailey T. Haug,
and John T. McLaughlin
Weapons and Materials Research Directorate, ARL

Abstract

Motion of the muzzle of a shoulder-fired weapon occurs during firing because of many factors, such as breathing, trigger pull, and flinching, and can also be introduced as a result of offhand shooting in less than stable shooting positions. This motion can have adverse effects on the capability of the weapon to hit a target because the shooter is unable to accurately position the muzzle of the weapon onto the target as the projectile exits the barrel. Electronic fire control offers the opportunity to extend the range and accuracy that can be achieved by a sniper or a long-range shooter with the implementation of a dynamic ballistic solution and a precise firing time selection. The Inertial Reticle Technology (IRT) has been developed at the U.S. Army Research Laboratory (ARL) as a novel solution to fire control for such applications. Therefore, to improve the accuracy of a shoulder-fired weapon, ARL applied the IRT to a rifle.

This report presents the complete details of how the IRT was applied to a Remington 700 sniper rifle, along with analysis of long-range live-fire test data fired by a test engineer and an Army sniper.

Table of Contents

		Page
	List of Figures	v
1.	Introduction	1
2.	The IRT Applied to a Weapon Fired From the Shoulder	2
3.	The IRT Applied to a Remington 700 Sniper Rifle	5
4.	Initial Testing of the IRT Applied to a Remington 700 Sniper Rifle	7
5.	Long-Range Testing of the IRT Applied to a Remington 700 Sniper Rifle at the Michaelsville Experimental Facility	9
6.	Long-Range Testing of the IRT Applied to a Remington 700 Sniper Rifle at the Romney Creek Experimental Facility	10
7.	Conclusions	11
8.	References	13
	Appendix: Computer Programs	15
	Distribution List	35
	Report Documentation Page	37

INTENTIONALLY LEFT BLANK.

List of Figures

Figure		Page
1.	Inertial Reticle Display of an E-Type Silhouette Target at a Range of 800 m	3
2.	Inertial Reticle Target Acquiring and Firing Sequence	3
3.	The IRT Applied to a Remington 700 Sniper Rifle (Right Side View)	6
4.	The IRT Applied to a Remington 700 Sniper Rifle (Left Side View)	6
5.	A Remington 700 Sniper Rifle With the IRT Being Fired From a Recoil Mount....	8
6.	A Remington 700 Sniper Rifle With the IRT Being Fired From a Shoulder Mount	8

INTENTIONALLY LEFT BLANK.

1. Introduction

The continuing evolution of smaller, low power, more capable computers and sensors has created the opportunity to explore the potential of electronic fire control for individual weapons and crew-served weapons where this technology has not been previously available. Electronic fire control offers the opportunity to extend the range and accuracy that can be achieved by a sniper or a long-range shooter with the implementation of a dynamic ballistic solution and a precise firing time selection. The Inertial Reticle Technology (IRT) has been developed at the U.S. Army Research Laboratory (ARL) as a novel solution to fire control for such applications. The origins of the IRT can be traced to investigations to improve the accuracy of helicopter munitions conducted by the U.S. Army Ballistic Research Laboratory, now ARL, in the early 1970s. This was the beginning of a series of programs focused on increasing the aiming accuracy of weapons in dynamic firing conditions where the weapon and platform performance were degraded by unwanted motion.

Motion of the muzzle of a shoulder-fired weapon occurs during firing because of many factors, such as breathing, trigger pull, and flinching, and can also be introduced as a result of offhand shooting in less than stable shooting positions. This motion can have adverse effects on the capability of the weapon to hit a target because the shooter is unable to accurately position the muzzle of the weapon onto the target as the projectile exits the barrel. Therefore, to improve the accuracy of a shoulder-fired weapon, ARL applied the IRT to a rifle. In 1991, initial experiments were conducted with an M16A2. The Special Operations Command initiated a research and development program, Advanced Sniper Weapon Fire Control System, known as Project White Feather, to look at the potential of extending the range of the sniper. The focus of the program was to develop a real-time crosswind sensor for sniper application. The IRT has been an integral part of the project as a result of the ability to dynamically integrate the crosswind and other environmental measurements into the ballistic solution and the ability to compensate for any residual motion of the shooter that becomes exaggerated in long-range shooting. The marksmanship skills of the military sniper reduce the unwanted motion of the weapon to extremely small angles; however, as the range is extended, even these small angles

can affect the accuracy. The IRT is a datum that cues the weapon to shoot when the shooter intends and not simply upon the release of the sear. To demonstrate the capabilities of the IRT to enhance the performance of a sniper weapon, the IRT was applied to a Remington 700, and this report documents the results obtained.

2. The IRT Applied to a Weapon Fired From the Shoulder

The IRT replaces the conventional sights or scope on the shoulder-fired weapon with a video camera and a video display that are mounted to the weapon. Two angular rate sensors are also mounted to the weapon that, when integrated, provide measurements of the angular displacements of the weapon in the elevation and the azimuth directions. Cant measurements are not made, because the weapon does not rotate significantly about its barrel centerline when fired from the shoulder. A small computer is used to generate two electronic pointers that are overlaid on the video image. The first pointer is a circle that is aligned with the barrel centerline of the weapon and thus represents the aim point. With inputs from other sensors, the aim point can be ballistically corrected for range, crosswind, relative elevation between the shooter and the target, and secondary effects, such as temperature and air density. If these inputs are available, the ballistic solution or predicted hit point can be calculated and displayed in real-time, eliminating the need for "Kentucky windage" (Von Walde and Metz 1999). The second pointer is a crosshair, referred to as the inertial reticle, and is driven in opposition to the weapon's angular displacements, as measured by the integrated rate sensors, so that the crosshair appears to remain fixed relative to the target, even though the weapon might be moving.

A typical display with the inertial reticle and the weapon aim point labeled is shown in Figure 1. Since there is no relative motion between the inertial reticle and the target, it is easy to position the inertial reticle over the desired target, using a joystick, even though the video scene might be moving around significantly. This is shown in the first frame of Figure 2. In the still frame, the dynamic nature of the display and the aiming process is not apparent, which makes it more difficult to appreciate the benefits of the reticle. In frame 1, the operator can place the

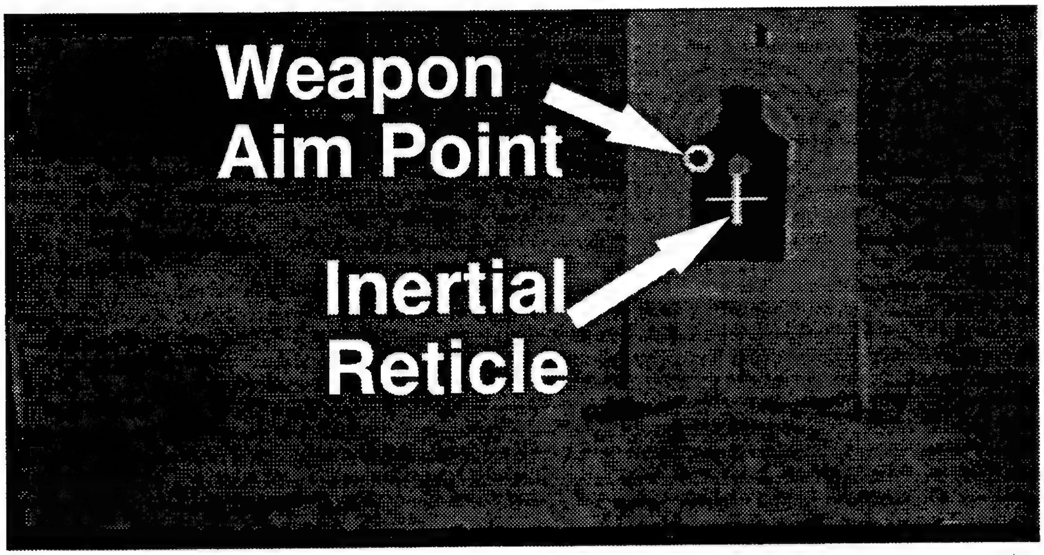
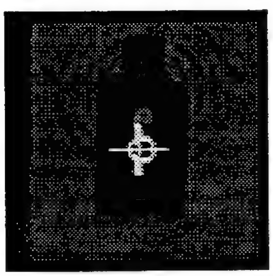
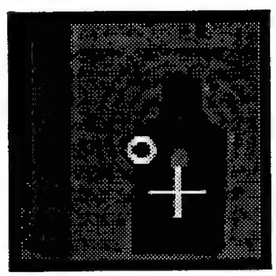


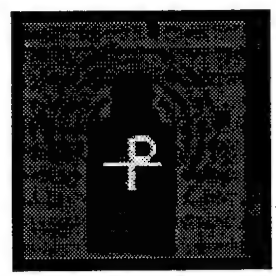
Figure 1. Inertial Reticle Display of an E-Type Silhouette Target at a Range of 800 m.



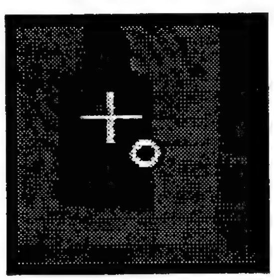
1. Place Reticle on Target



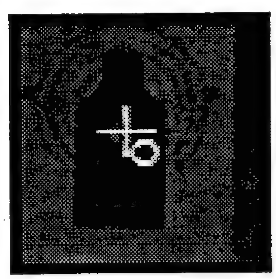
2. Inertial Reticle Placed on Target



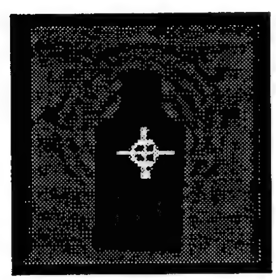
3. Inertial Reticle Remains Fixed



4. Weapon Motion Drives Aim Point on Display



5. Weapon Moved to Align Aim Point With Inertial Reticle



6. Weapon Fires

Figure 2. Inertial Reticle Target Acquiring and Firing Sequence.

inertial reticle on the target without the discipline required to squeeze off the shot. Two means of placing the reticle on the target are provided. The reticle can be driven with a joystick, as was done in the experiments with the M16A2. The reticle can also be locked to the bore sight, to allow the weapon to be used to move the reticle. This is the implementation chosen for the sniper application, because the sniper provides a stable support for the weapon, allowing the reticle to be quickly placed on the target. Once the inertial reticle is accurately placed over the desired target, the computer uses continuous measurements of the inertial reticle's angular position relative to the aim point to control the precise firing time. This is used to ensure that the exit of the projectile from the barrel occurs when the muzzle of the weapon is positioned over the target. As shown in frames 4 and 5 of Figure 2, if the operator is satisfied with the position of the inertial reticle, the weapon is moved to align the aim point, the ballistic solution, with the inertial reticle. The operator enables the weapon to fire, and the computer will select the firing time such that the bullet leaves the muzzle as the aim point crossed the reticle. The operator remains in control of the firing of the weapon, but the computer provides an electronic trigger for precise control of the firing time. The firing is accomplished by means of an electrical solenoid that is attached to the trigger of the weapon.

The application of the IRT to a rifle required small, relatively inexpensive inertial sensors. The large gyroscopic sensors that were used for the initial experiments were hardly the size or cost required for this application. The sensors must survive the firing impulse, be lightweight, and be small enough to be mounted on the weapon. Ring laser gyros (RLG) were considered for this application, but were rejected because of the cost and the size of the RLG required. A fiber optic gyro (FOG) is smaller and lighter than an equivalent RLG, but the cost of a FOG at the time was still too high to be considered. The silicon micro-machined gyros are very small, lightweight, and low cost, but at the time, did not have the accuracy required for an IRT application. The Systron Donner quartz rate sensors (QRS-11-00100-101) were selected, and a mount was designed to hold the sensors orthogonally on the rifle to measure angular velocities in elevation and azimuth. This sensor operates on ±5 V and draws about 0.8 W. It has a bias stability of about 7°/hr and a random walk of about $0.15°/\sqrt{\text{hr}}$. The bias can be continuously adjusted by the computer to minimize the drift that occurs in the inertial reticle when the angular velocity is integrated to get displacement. The random walk, however, cannot be corrected and

causes the inertial reticle to move around on the scene, eventually moving away from the target. A decrease in the random walk would make a major improvement in the stability of the inertial reticle.

3. The IRT Applied to a Remington 700 Sniper Rifle

The IRT was applied to a Remington 700 sniper rifle, as shown in Figures 3 and 4. The Remington 700 sniper rifle was fitted with a Cast Glance Video Riflescope. The Cast Glance Video Riflescope is a Leupold MK4 M-3 scope that has been modified to accept a beam splitter and a Cast Glance black-and-white video camera. The Cast Glance black-and-white video camera captures less than 50% of the image from the scope, thus when combined with the 10× of the Leupold MK4 M-3 scope, the video system gives an image of over 20×. The Cast Glance Video Riflescope can also be used as a conventional 10× scope by viewing through the rear of the scope. The video image from the Cast Glance Video Riflescope was viewed by the shooter on a Sony Flat Display Monitor (FMD-030) that was attached to the weapon. Two Systron Donner quartz rate sensors (QRS-11-00100-101) were mounted beneath the weapon. Also mounted beneath the weapon, behind the quartz rate sensors, was an electrical solenoid that was attached to the trigger.

Control of positioning the inertial reticle over the target was accomplished by means of a thumb switch that was mounted on top of the stock behind the bolt. By depressing the thumb switch and holding it depressed, the inertial reticle and the aim point circle were aligned, and by moving the weapon, both the inertial reticle and the aim point circle could be accurately positioned over the desired target. Then, by releasing the thumb switch, the inertial reticle was positioned over the target, where it remained unaffected by any subsequent weapon motion. The inertial reticle could also be easily placed over the desired target by using the joystick. Theoretically, once the inertial reticle is positioned accurately over the target, it should stay there indefinitely. However, due to random walk in the quartz rate sensors the inertial reticle would drift off from the target after about 3 s. The 3 s was more than ample time to allow the shooter to fire the weapon. If more time was needed, all the shooter had to do was depress the thumb switch and reacquire the target or use the joystick to reposition the inertial reticle over the target.

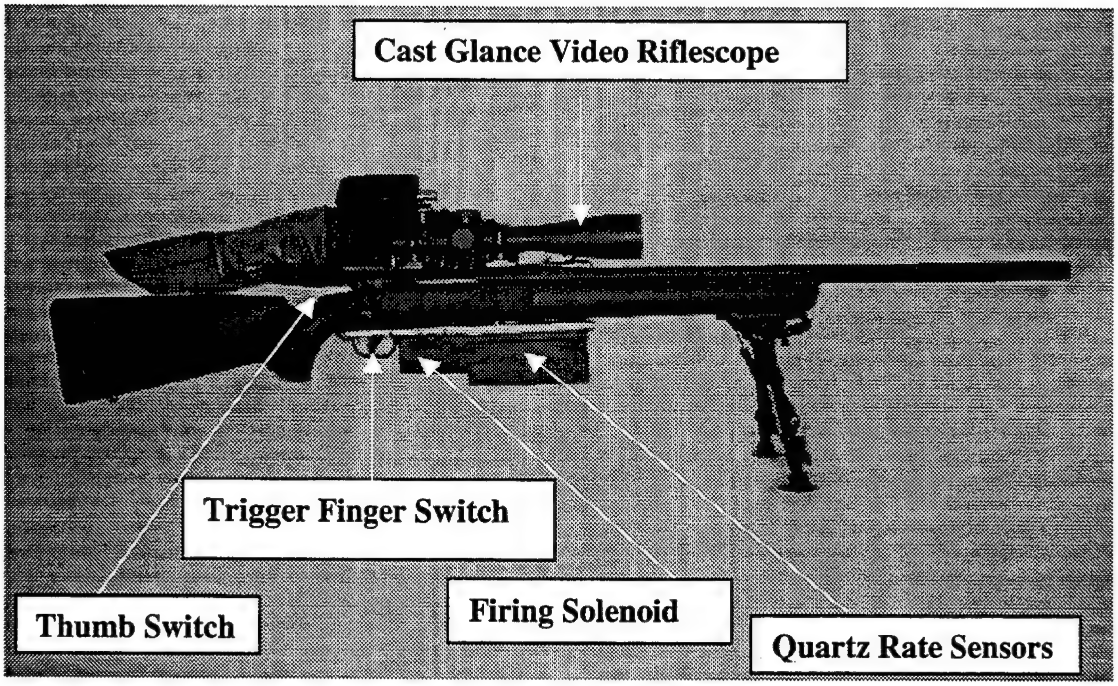


Figure 3. The IRT Applied to a Remington 700 Sniper Rifle (Right Side View).

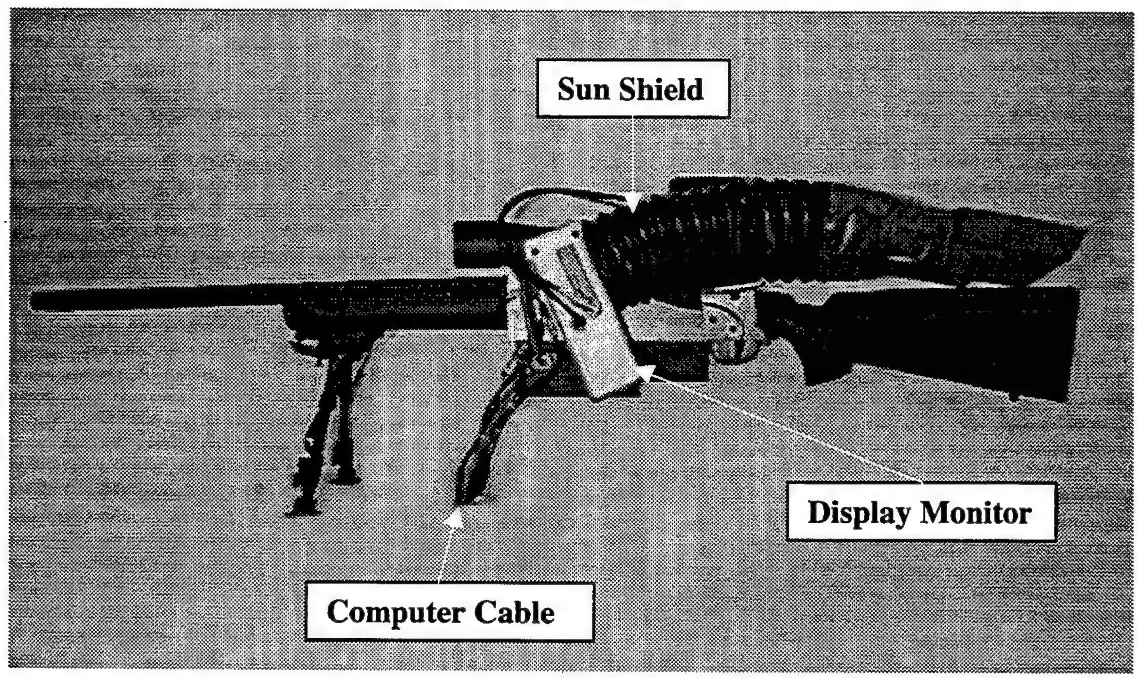


Figure 4. The IRT Applied to a Remington 700 Sniper Rifle (Left Side View).

Control of firing the weapon was accomplished by means of a trigger finger switch that was mounted to the trigger. If the shooter was satisfied with the position of the inertial reticle over the desired target, then, by depressing the trigger finger switch and holding it depressed, the electrical firing solenoid was enabled and the shooter simply moved the weapon to align the aim point circle with the inertial reticle. The weapon automatically fired when the aim point circle was aligned with the inertial reticle, and the projectile exit from the barrel occurred when the muzzle of the weapon was positioned over the target. The integration of the quartz rate sensor signals, the generation of the predictive fire control algorithm, and the firing of the weapon was accomplished by a small WinSystems 486 SLC computer and power supply that were attached to the weapon by means of a light cable. The generation of the electronic pointers was accomplished by a small 386 SX computer that was fed directly into the WinSystems 486 SLC computer. No attempt will be made to reduce the size of the computer and electronic hardware until the development of the crosswind sensor is completed. Integrating the crosswind sensor with the IRT could easily be accomplished with a single computer handling all the computational requirements. The complete computer programs for both of the computers are presented in the Appendix.

4. Initial Testing of the IRT Applied to a Remington 700 Sniper Rifle

Prior to any long-range testing of the IRT applied to a Remington 700 sniper rifle, extensive short-range testing was done in the indoor experimental facility in Building 390 at ARL. Over 100 rounds were fired from a recoil mount and from a shoulder mount to determine if the video camera, the beam splitter, the video display, and the quartz rate sensors could withstand the shock from firing. The recoil mount and the shoulder mount used in the initial firings are shown in Figures 5 and 6. Accuracy measurements were also taken during the initial testing for each round fired. The accuracy acceptance specification for M80 ammunition, which was fired in the initial testing, converts to an average standard deviation of .23 mil in both the elevation and the azimuth directions. The average standard deviations in the elevation and the azimuth directions

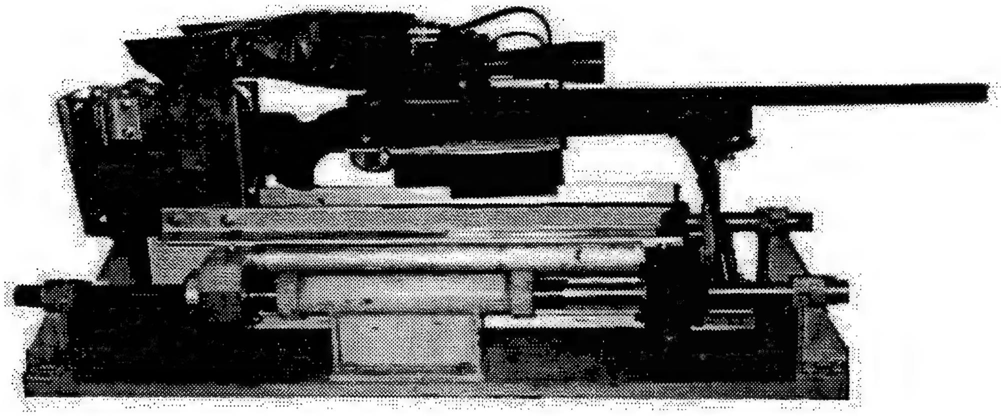


Figure 5. A Remington 700 Sniper Rifle With the IRT Being Fired From a Recoil Mount.

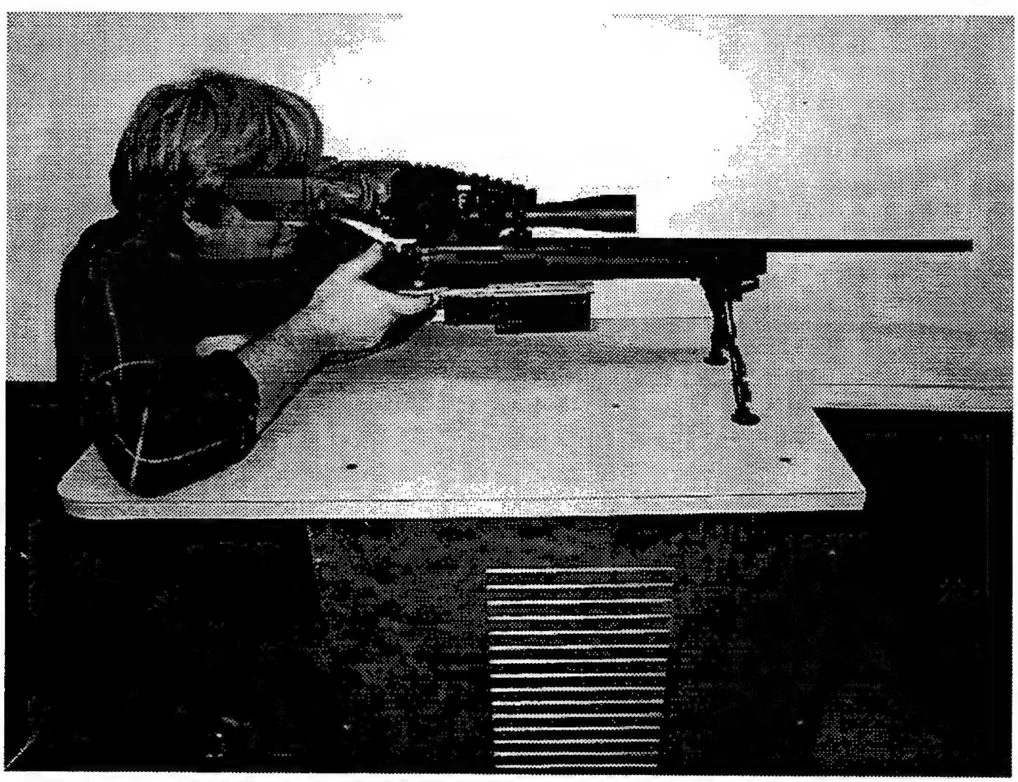


Figure 6. A Remington 700 Sniper Rifle With the IRT Being Fired From a Shoulder Mount.

for several 10-round groups fired from the recoil mount were .19 mil and .21 mil, respectively. The average standard deviations in the elevation and the azimuth directions for several 10-round groups fired from the shoulder mount were .22 mil and .21 mil, respectively. Since the average standard deviations for the recoil mount firings and the shoulder mount firings were essentially the same as the accuracy acceptance specification, it was felt that the IRT applied to a Remington 700 sniper rifle was achieving its maximum possible increase in accuracy when fired from the shoulder mount and no further short-range indoor testing was done. There was also no damage to the video camera, the beam splitter, the video display, or the quartz rate sensors after firing over 100 rounds.

5. Long-Range Testing of the IRT Applied to a Remington 700 Sniper Rifle at the Michaelsville Experimental Facility

After the initial testing was completed, long-range testing of the IRT applied to a Remington 700 sniper rifle was done at the Michaelsville experimental facility at the U.S. Army Aberdeen Test Center (ATC) with both a test engineer and an Army sniper firing from the shoulder mount that was used in the previous short-range initial testing. The maximum range for all of the long-range testing was 800 m because beyond 800 m, the round of ammunition being fired becomes subsonic and dispersion increases, which would bias the test results. The Michaelsville experimental facility used for the long-range testing is closed out to 500 m, so the effects of crosswind are minimal on this experimental facility. The accuracy acceptance specification for M118 LC ammunition, which was used in the long-range testing because it is more accurate than M80 ammunition, converts to an average standard deviation of .16 mil in both the elevation and the azimuth directions. The average standard deviations in the elevation and the azimuth directions for several five-round groups of M118 ammunition fired from the shoulder mount by the test engineer were .17 mil and .15 mil, respectively. The average standard deviations in the elevation and the azimuth directions for several five-round groups of M118 LC ammunition fired from the shoulder mount by the Army sniper were .16 mil and .13 mil, respectively. Since the average standard deviations for the shoulder mount firings by the test engineer and the Army

sniper were essentially the same as the accuracy acceptance specification, it was felt that the IRT applied to a Remington 700 sniper rifle was achieving its maximum possible increase in accuracy when fired from the shoulder mount.

6. Long-Range Testing of the IRT Applied to a Remington 700 Sniper Rifle at the Romney Creek Experimental Facility

After the long-range testing at the Michaelsville experimental facility was completed, long-range testing of the IRT applied to a Remington 700 sniper rifle was done at the Romney Creek experimental facility at ATC, with a test engineer firing from the shoulder mount that was used in the previous short-range and long-range testing. At the same time, long-range testing was also done with an Army sniper firing a standard Remington 700 sniper rifle with a Leupold 10× scope from a supported prone position on the ground. The maximum range for all of the long-range testing was again 800 m. The experimental facility at Romney Creek is open, and the effects of the crosswind can be significant on this facility.

The objectives of this long-range testing were to determine if the IRT applied to a Remington 700 sniper rifle could automatically correct for the effects of the crosswind and to determine how well an Army sniper could correct for the effects of the same crosswind using his sniper training techniques. In the testing with the IRT applied to a Remington 700 sniper rifle, four sets of anemometers and wind direction indicators were placed downrange along the path of the projectile flight to the target. From the anemometers and the wind direction indicators, a continuous measurement of the wind velocity and direction over the entire length of the 800-m range was inputted into the computer of the IRT and a continuous crosswind correction was calculated and used to drive the aim point circle automatically without any shooter action. The aim point circle was simply moved to the right or left depending on the crosswind correction.

In the testing with the Army sniper using the standard Remington 700 sniper rifle with the Leupold 10× scope, the Army sniper was not told the crosswind conditions but had to estimate them using his sniper training techniques to determine the correction for the crosswind. He fired

the complete 10-round group without knowing where the projectiles hit the target until after firing the tenth round. The standard deviations and the centers of impact in the elevation and the azimuth directions for a 10-round group of M118 LC ammunition fired from the shoulder mount by the test engineer using the IRT applied to a Remington 700 sniper rifle were .21 mil and .19 mil, respectively, for the standard deviations, and −4 cm and 7 cm, respectively, for the centers of impact. The standard deviations and the centers of impact in the elevation and the azimuth directions for a 10-round group of M118 LC ammunition fired from the supported prone position on the ground by the Army sniper using a standard Remington 700 sniper rifle with a Leupold 10× scope were .18 mil and .22 mil, respectively, for the standard deviations, and 4-cm and 36-cm, respectively, for the centers of impact. Since a standard E-type silhouette target is about 50-cm wide, the center of impact in the azimuth direction for the 10-round group fired by the Army sniper was off to the right of the target by 11 cm.

7. Conclusions

(1) The IRT applied to a Remington 700 sniper rifle improved the accuracy to such an extent that a test engineer was able to keep five-round groups of M118 LC ammunition within a 50-cm circle, which was centered on the target, while firing at a range of 800 m, which is about the accuracy limitation of the ammunition itself when fired from a fixed mount at a 800-m target.

(2) The IRT, with crosswind correction input, applied to a Remington 700 sniper rifle, improved the accuracy to such an extent that a test engineer was able to keep a 10-round group of M118 LC ammunition on an E-type silhouette target at a range of 800 m. At the same time and on the same range, an Army sniper firing his own weapon, using sniper training techniques for estimating the crosswind correction, was not able to keep a 10-round group of M118 LC ammunition on an E-type silhouette target at a range of 800 m. Note that Army snipers usually operate in a two-man team with one man aiming and firing the weapon and the second man calling out crosswind corrections. If the second man would have been used in these tests to call out the crosswind corrections, then, quite possibly, the Army sniper would have been able to keep the 10-round group

on the target. The bulk of the sniper's time is spent as observers and/or scouts. However, "taking the shot" successfully is that upon which the success of the mission—as well as the sniper's life—may hang. However, in the same manner with the IRT, a second man with his own display and joystick could be used to continually keep the inertial reticle on the target. This would allow the man pointing and firing the weapon to only concentrate on making the aim point circle cross the inertial reticle. Also, with this two-man IRT team sharing the same field of view, target identification is greatly facilitated.

8. References

Von Walde, Raymond, and Dennis Metz. "Sniper Weapon Fire Control Error Budget Analysis." ARL-TR-2065, U.S. Army Research Laboratory, Aberdeen Proving Ground, MD, August 1999.

INTENTIONALLY LEFT BLANK.

Appendix:

Computer Programs*

* Program Reticle is the computer program for the 386 SX computer. Program IRS4 is the computer program for the WinSystems 486 SLC computer.

INTENTIONALLY LEFT BLANK.

```
Program Reticle; {May 3rd, 1994, 09:54 A.M., November 3, 1994}
  uses                    {Video Charlie}
    graph,  crt;
  type a10 = array[0..9] of word;

    var      A          :a10;
         Gd, Gm          :integer;
         p0, p1         :boolean;
         ccc,ddd         :byte;
         vvv,www          :word;
    x11,x12,x21,x22,y11,y12,y21,y22:word;
    a11,b11:word;

(****************** ******************)
Procedure Initialize;
begin
 asm
  mov dx,74bh
  mov al, 09bh  {Ports A, B and C all input}
  out dx,al
 end;
;
 x11:=0; x12:=0; y11:=0; y12:=0; x21:=0; x22:=0; y21:=0; y22:=0;
 a11:=320; b11:=100; a[0]:=0; a[1]:=0; a[2]:=0; a[3]:=0;
end;


(****************** ******************)
Procedure GetData(var ddd,ccc:byte; var vvv,www:word; var p0,p1:boolean);
begin
 {sxxx xxxx xccc cddd  dddd dddd}

asm  {$G+}
    mov dx,749h
@0:              in al,dx     ;  mov cl,al  {xccc cddd > cl}
    dec dx    ;  in al,dx     ;  mov ch,al  {dddd dddd > ch}
    inc dx    ;  in al,dx                {xccc dddd > al}
    and al,cl  ;  and al,080h  ; jz @0

    les bx,DDD
    mov al,cl                       {xccc cddd > al}
    ror al,3                       {dddx cccc > al}
    and al,0fh                       {0000 cccc > al}
    mov es:[bx],al                     {0000 cccc > DDD}
```

```
;
    les bx,VVV  ;  mov es:[bx],ch              {dddd dddd > VVV}
    mov al,cl  ;  and al,07h    ;            {0000 0ddd > al}
    mov es:[bx+1],al                       {0000 0ddd > vvv+1}
;
    les bx,CCC
    mov dx,074ah
    in al,dx
    mov es:[bx],al
end;


p1 := p0;
if ((CCC and 128) > 0) then  p0 := true else p0 := false;

if ( (p0 = true) and (p1 = false)  ) then
 begin
  A[DDD] := VVV;

  If (DDD and 2)=0 then
   begin
     SetColor(Black);
     line(x11,y11,x12,y12);
     line(x21,y21,x22,y22);
     line(x21-2,y21,x22-2,y22);
     line(x21+2,y21,x22+2,y22);

     VVV := A[0];
     if vvv > 638 then vvv := 638;
     x11 := vvv - 20; if x11<  1 then x11:=  1;
     x12 := vvv + 20; if x12> 638 then x12:= 638;
     x21 := vvv; x22 := vvv;

     WWW := A[1];
     if www > 198 then www := 198;
     y21 := www -  8; if y21<  1 then y21:=  1;
     y22 := www +  8; if y22> 198 then y22:= 198;
     y11 := www; y12 := www;

     SetColor(White);
     line(x11,y11,x12,y12);
     line(x21,y21,x22,y22);
     line(x21-2,y21,x22-2,y22);
     line(x21+2,y21,x22+2,y22)
   end
```

```
else
  begin
    SetColor(Black);
    circle(a11,b11,8);  circle(a11,b11,9);

    VVV := 315;
     {A[2]; Restore for wind, July 21, 1995}
    if vvv > 630 then vvv := 630;
    if vvv <   8 then vvv :=   8;
    a11 := vvv;

    WWW := 98; {A[3]; Restore for wind, July 21, 1995}
    if www > 190 then www := 190;
    if www <   8 then www :=   8;
    b11 := www
  end; {if(ddd and 2 = 0}

    SetColor(White);
    Circle(a11,b11,8); Circle(a11,b11,9);


 end;  {if ( (p0 = true) and (p1 = false) )}

end;   {procedure}


(* * * * * * * * * * * * * * * * * *  * * * * * * * * * * * * * * * * * * * *)
  begin  {Main}

   Gd:=detect;
   Initgraph (Gd, Gm, 'c:\tp\bgi');
   if graphresult <> grOk then
    begin
      writeln('Cannot file graphics files.  Press any key to continue.');
      readln; halt (1);
    end;


(*------ initial values ------*)

Initialize;     p0:= true;

while keypressed = false do  GetData(ddd,ccc,vvv,www,p0,p1);
     closegraph;
 end.
```

```
Program IRS4;  {September 14, 1992, June 21, 1993, March 24, 1994}
{N+}        {March 31, 1994 - Predict 40 ms, cycle 2 ms}
Uses        {April 25, 1994 - used with video reticle genrator}
        {1:25 P.M., September 11,1995}

    Graph,Crt;          {Port assignments: 050h to 057h}
                    {Port assignments:  80  to  87 }
Const Size = 20;    Half  = 0.5;
type  seal = single;
    r33 = array[1..3,1..3] of real; i88 = array[0..7,0..7] of byte;
    r8 = array[0..7] of seal;
    i8 = array[0..7] of byte;
    rsize= array[1..size] of seal;


var
                        nnn  : i8;
                        zzz  :real;
B1,B2, dt, sum1,  sum2, wx,   wy,   wz :seal;  {Body fixed angular rates}
Offset1,Offset2, P1, P2, PrevErr,Tim     :seal;
ww,  channel,p6,  n3,   n2,   n1,   n0 :byte;  {Raw sensor inputs}
 port6, kkk,    n7,   n6,   n5,   n4 :byte;  {Raw sensor inputs}
      flop,      j, k, cnt, P7In      :byte;  {couters and flag}
                         KK :integer;
              Delay,   wrd    :Word;
                         nn :i88;     {Raw sensor inputs}
                      l,    i :integer;  {Loop counter}
   iii,jjj,        LLL,   Count   :word;
   sign,      Firp,   Fir,   error : boolean;
wx_store,wz_store, Theta1, Theta2      : rsize;
                    Time,w,wnd : seal;
                       head  : integer;
                      swr,swr1:byte;
(*------------------------------------------------------------------------*)
Procedure DAC(var channel:byte; var value:word);
begin
 asm
 {$G+}     {sccc cddd dddd dddd}

 mov dx,0195h       {195h > dx}
 and al,0  {Reset the data ready bit}
 out dx,al

 dec dx          {194h > dx}
 les bx,value;   mov ax,es:[bx];    {ah:xxxx xddd, al:dddd dddd}
```

```
  out dx,al;       {dddd dddd}        {Bits 0 - 7 > 194h}

  inc dx          {195h > dx}
                        {ah:xxxx xddd}
  and ah,7                 {ah:0000 0ddd}
  les bx,channel;  mov al,es:[bx];    {al:xxxx cccc}
  rol al,3;                {al:xccc cxxx}
  and al, 078h;              {al:0ccc c000}
  or al,ah;                {al:0ccc cddd}
  or al,080h                {al:1ccc cddd}
  out dx,al
  end;
end;


(*---------------------------------------------------------------------*)

Procedure GetPort7( var A:Byte);
begin
 asm
  mov dx,0197h;
  in al,dx
  les bx, A; mov es:[bx],al;
 end;
end;



(*---------------------------------------------------------------------*)

Procedure SetPort6_7( var A:Byte);
begin  asm
       -mov dx,0196h;
les bx, A; mov al,es:[bx];  { (A) > al}
        or al,080h;
        mov es:[bx],al;
        out dx,al
end; end;

(*---------------------------------------------------------------------*)

Procedure ResetPort6_7( var A:Byte);
begin  asm
        mov dx,0196h;
```

```
les bx, A; mov al,es:[bx];
        and al,07fh;
        mov es:[bx],al;
        out dx,al
end; end;

(*-------------------------------------------------------------------*)

Procedure SetPort6_0( var A:Byte);
begin  asm
        mov dx,0196h;
les bx,A;  mov al,es:[bx];
        or al,1;
        mov es:[bx],al;
        out dx, al
end; end;


(*-------------------------------------------------------------------*)
Procedure SetPort6_1( var A:Byte);
begin  asm
        mov dx,0196h;
les bx,A;  mov al,es:[bx];
        or al,2;
        mov es:[bx],al;
        out dx, al
end; end;


(*-------------------------------------------------------------------*)

Procedure ResetPort6_0( var A:Byte);
begin  asm
        mov dx,0196h;
les bx,A;  mov al,es:[bx];
        and al,0feh;
        mov es:[bx],al;
        out dx,al
end; end;


(*-------------------------------------------------------------------*)
```

```
Procedure ResetPort6_1( var A:Byte);
begin  asm
        mov dx,0196h;
les bx,A;  mov al,es:[bx];
        and al,0fdh;
        mov es:[bx],al;
        out dx,al
end; end;


(*----------------------------------------------------------------------*)

Procedure Output( Channel:byte; var A:seal);
begin
      if a > 639 then wrd := 639
   else if a <  0 then wrd :=  0
   else              wrd :=  trunc(a);
   DAC (channel, Wrd);
end;



(*----------------------------------------------------------------------*)
Procedure IRS
(
   var  sx,sz,bx,bz,px,pz      :seal; var Fir:Boolean
 );
var a :byte;    wd:seal;
begin

 swrl := swr;
 swr := P7In and 2;
 if ( (swrl = 2) and (swr = 0)) then wnd := w;
 if swr = 0 then wd := wnd else wd := w;
 if swr = 0 then begin sx := bx; sz := bz + wd          end;
 if P7In and 4 = 0 then begin bx := sx; bz := sz end;



 if (  (Fir = False) and (P7In and 1 = 0)  ) then
    begin
 if ((abs(bx-px) <0.6) and (abs(bz + w - pz) < 1.5) ) then Fir:= True;

    end;
```

```
end;

(*----------------------------------------------------------------------*)
Procedure Convert(var s:real; var nn:i8);
begin

 {real: s:1  f:39   e:8. v := (-1)**s*2**(e-129)*(1.f).  if e=0 then v:=0}
 {     b47  b46-b8  b7-b0  }
    asm          {MSByte  ah, al, ch, cl  LSByte}
{$G+}
    les bx,nn
    mov al,es:[bx+4]   { al: s.in m.2 m.1 m.0  x   x  b25  b24}
    and al,3           { al:  0  0  0  0  0  0 b25  b24}

    mov dl,es:[bx+5]   { dh: s.in m.2 m.1 m.0  x  x  b27  b26}
    and dl,3           { dh:  0  0  0  0  0  0 b27  b26}
    rol dl,2
    or al,dl           { al:  0  0  0  0 b27 b26 b25 b24}

    mov dl,es:[bx+6]   { dl: s.in m.2 m.1 m.0  x  x  b29  b28}
    and dl,3
    rol dl,4
    or al,dl

    mov dl,es:[bx+7]   { dl: s.in m.2 m.1 m.0  x  x  b31  b30}
    and dl,3
    rol dl,6
    or al,dl           {al:  b31 b30 b29 b28 b27 b26 b25 b24}



    mov cl,es:[bx+2]   {cl: b23 b22 b21 b20 b19 b18 b17 b16}

    mov dh,es:[bx+1]   {dh: b15 b14 b13 b12 b11 b10  b9  b8}

    mov dl,es:[bx+0]   {dl:  b7  b6  b5  b4  b3  b2  b1  b0}

    mov ch,0           {ch:  0  0  0  0  0  0  0  0}

    test al,128        {al: b31  0  0  0  0  0  0  0}
    jz @00             {Jump if negative}

      {sign:1, al:8, cl:8, dh:8, dl:8, 00:8, ah:8}
    xor al,255    {b31 b30 b29 b28 b27 b26 b25 b24}
```

```
    xor cl,255    {b23 b22 b21 b20 b19 b18 b17 b16}
    xor dh,255    {b15 b14 b13 b12 b11 b10 b09 b08}
    xor dl,255    {b07 b06 b05 b04 b03 b02 b01 b00}
    add dl,1;   adc dh,0;   adc cl,0;   adc al,0;   mov ch,128

@00: mov ah,128+32   {sign:1, al:8, cl:8, dh:8, dl:8, 00:8, ah:8}
    test al,255      {Check for all zeros}
    jnz @3

    mov al,cl ; mov cl,dh ; mov dh,dl ; mov dl,0 ; mov ah,128+8+8+8
    test al,255
    jnz @3

    mov al,cl ; mov cl,dh ; mov dh,00 ;          mov ah,128+8+8
    test al,255
    jnz @3

    mov al,cl ; mov cl,00 ;                      mov ah,128+8
    test al,255
    jnz @3

    mov al,00 ;                        ; mov ah,0
    jmp @1          {Finished}


@3:
                      test al,128;   jnz @1
     dec ah;            test al,64;    jnz @56
     dec ah;            test al,32;    jnz @55
     dec ah;            test al,16;    jnz @54
     dec ah;            test al,8;     jnz @53
     dec ah;            test al,4;     jnz @52
     dec ah;            test al,2;     jnz @51
     dec ah
;               {al, cl, dh, dl, 00, ah}
@50:
     rcl dl,1
     rcl dh,1
     rcl cl,1
     rcl al,1

@51:
     rcl dl,1
     rcl dh,1
```

```
    rcl cl,1
    rcl al,1
@52:
    rcl dl,1
    rcl dh,1
    rcl cl,1
    rcl al,1
@53:
    rcl dl,1
    rcl dh,1
    rcl cl,1
    rcl al,1
@54:
    rcl dl,1
    rcl dh,1
    rcl cl,1
    rcl al,1
@55:
    rcl dl,1
    rcl dh,1
    rcl cl,1
    rcl al,1
@56:
    rcl dl,1
    rcl dh,1
    rcl cl,1
    rcl al,1

@1:
    and al,127
    or al,ch

    les bx,s
    mov es:[bx+5],al
    mov es:[bx+4],cl
    mov es:[bx+3],dh
    mov es:[bx+2],dl
    xor al,al
    mov es:[bx+1],al
    mov es:[bx+0],ah

  end; {Asm}

end; {Proc Convert}
```

```
(*--------------------------------------------------------------------*)

Procedure Ack_Lo(var CNT:byte);
begin
 asm    {$G+}
  mov dx,0191h
  mov ah,0
@0: dec ah;   jz @1
  in al,dx;   test al,128 {bit7};   jnz @0
@1: les bx,CNT;   mov es:[bx],ah
 end; {Asm}
end;


(*--------------------------------------------------------------------*)
Procedure Ack_Hi(var CNT:byte);
begin
 asm    {$G+}
  mov dx,0191h
  mov ah,0
@0: dec ah;   jz @1;
  in al,dx;   test al,128 {bit7};   jz @0
@1: les bx,CNT;   mov es:[bx],ah
 end; {Asm}
end;


(*--------------------------------------------------------------------*)
Procedure Get_Result(var a1,a0:byte);
begin
 asm
  mov dx,0190h;   les bx,a0;   in al,dx;   mov es:[bx],al

  mov dx,0191h;   les bx,a1;   in al,dx;   mov es:[bx],al
 end;
end;


(*--------------------------------------------------------------------*)
Procedure Ext_Sign_Bit(var a0:byte);
begin
 asm       {$G+}
  les bx,a0
  mov al,es:[bx]
```

```
  and al,15
  test al,8
  jz @0
  or  al, 240
@0:
  mov es:[bx],al
 end;
end;


(*---------------------------------------------------------------------*)
Procedure Int(var n2:byte; var n4,n3:byte);
begin
 asm
 {$G+}
 les bx,n2; mov al,es:[bx]; mov ah,al;

  and al, 112 {Mask channel}; ror al,4;  les bx,n4; mov es:[bx],al;

  mov al,ah;
  and al,12 {Mask set};      ror al,2;  les bx,n3; mov es:[bx],al
 end; {asm}
end;

(*---------------------------------------------------------------------*)
Procedure C16(var count:word; var n0,n1:byte);
begin
 asm
 {$G+}
 les bx,n0; mov al,es:[bx];   les bx,n1; mov ah,es:[bx];
 les bx,count; mov es:[bx],ax
 end; {asm}
end;

(*---------------------------------------------------------------------*)
procedure get(var n0:byte);
begin
 asm     {$G+}
  mov dx,0191h
  in al,dx
  les bx,n0
  mov es:[bx],al
 end; {Asm}
end;
```

```
(*----------------------------------------------------------------------*)
 Procedure GetReading
(
  var nn        :I88;
  var Count     :Word;
  var n4,n5     :Byte
);
var j,k,n3,n2,n1,n0,CNT :Byte;
 begin {l 1}
  error := false;
 j:= 0;
 n5:=0;
 while  (j < 7) do
  begin  {j 2}
       inc(n5);

       k := 0;
       n4 := 0;
       while (k<3) do
         begin  {k 3}
           inc (n4);
           if (( n2 and 128) = 0) then
             begin   {0 4}
                setport6_0(Port6);
                ack_hi(CNT);          {Wait for ack to go high}
                if (CNT =0) then  exit;
                get_result (n2,n0);
                int(n2,j,k);
                nn[j,k+4] := n2;
                nn[j,k  ] := n0
             end    {0 4}
         else
             begin  {1 4}
                resetPort6_0(Port6);
                ack_lo(CNT);          {Wait for ack to go low}
                if (CNT =0) then exit;
                get_result (n2,n1);
                int(n2,j,k);
                nn[j,k+4] := n2;
                nn[j,k  ] := n1 ;
             end;   {1  4}

         end;  {k 3}

{    if n4 <> 4 then error := true;}
```

```
  end;   {j 2}
  setPort6_0(Port6);
  resetPort6_0(Port6);

  if n5 <> 8 then error := true;

  c16(count,nn[6,3],nn[7,3]);

end;


(*---------------------------------------------------------------------------*)
Procedure Sensors (var wx,wz:seal);
var j,k:byte;
begin
   GetReading (     nn, Count, n4,n5    );
      j := 3;
      if (error=false) then begin
      for k := 0 to 7 do nnn[k] := nn[j,k]; Convert(zzz,nnn);end;
      wx := zzz;

      j  := 7;
      if (error=false) then begin
      for k := 0 to 7 do nnn[k] := nn[j,k]; Convert(zzz,nnn);end;
      wz := zzz;
end;

(*---------------------------------------------------------------------------*)
Procedure Inte(var sum1,sum2,OffSet1,Offset2:seal);
var A,B:seal;wxx,wzz:seal;
begin
   Sensors(wx,wz);


   tim := 0.04;
   wxx:= wx + offset1 * count;
      if wxx >0 then offset1 := offset1 - tim
      else offset1 := offset1 + tim;
   wzz := wz + offset2 * count;
      if wzz > 0 then offset2 := offset2 - tim
      else offset2 := offset2 + tim;
;
   sum1 := sum1 + wxx *0.00030;    {Vertical integration constant}
   sum2 := sum2 - wzz *0.00080;    {Horizontal integration constant}
;
```

```
    if flop <= 1 then begin  A := -sum1+100; OutPut(1,A); flop := 2 end
  else if flop =  3 then begin  A := -B1  +100; OutPut(3,A); flop := 4 end
  else if flop =  2 then begin  B :=  sum2+320; OutPut(0,B); flop := 3 end
  else if flop >= 4 then begin  B :=  B2+w+320; OutPut(2,B); flop := 1 end;
;
end;


Procedure DAC00(var xx:integer); begin asm {$G+}
    mov dx,0ffe0h; les bx,xx; mov ax,es:[bx]; xor ah,008h
    out dx,ax end end;

Procedure DAC01(var xx:integer); begin asm {$G+}
    mov dx,0ffe2h; les bx,xx; mov ax,es:[bx]; xor ah,008h
    out dx,ax end end;

Procedure DAC02(var xx:integer); begin asm {$G+}
    mov dx,0ffe4h; les bx,xx; mov ax,es:[bx]; xor ah,008h
    out dx,ax end end;

Procedure DAC03(var xx:integer); begin asm {$G+}
    mov dx,0ffe6h; les bx,xx; mov ax,es:[bx]; xor ah,008h
    out dx,ax end end;



(*-------------------------------------------------------------------------*)
Procedure Predict(
              var S0,T0 :seal;
              var The1,The2 :rsize;
              var P,Q :seal);

var I,J,ss,pp,tt,qq:integer;  S1,S2,T1,T2,T,S:seal;
begin

The1[Head] := S0; The2[Head] := T0;
;
wx_store[head] := wx ; wz_store[head] := wz;
I := Head - 10; if I < 1 then I := I + Size;
S2 := the1[I]; T2 := the2[I];


P := S0 + 1*(S0 - S2);
Q := T0 + 1*(T0 - T2);
```

```
inc(Head) ;  if Head > Size then Head := 1;

if fir then P := P + 10;
if fir then Q := Q + 50;

ss := trunc(S0*16);
pp := trunc(P*16);
tt := trunc(T0*4);
qq := trunc(Q*4);

DAC00(ss); DAC01(pp); DAC02(tt); DAC03(qq);

end;

(*-----------------------------------------------------------------------*)
Procedure GetP2(var s:byte);
begin
asm {$G+}
 mov dx,0192h
 in al,dx
 les bx,s
 mov es:[bx], al;

 end;
end;



(*--------------------------------MAIN-----------------------------------*)

begin {main}
Head := 1; {Pointer to theta array}
for i := 1 to size do begin theta1[i]  := 0;  theta2[i] := 0 end;
PrevErr := 1e10;
sensors(wx,wz);
get_result (n2,n0);
Ack_Lo(CNT); {Test slave's output - low normal state}
if (CNT = 0) then
  begin
    writeln( 'Slave in wrong state.  Press any key');
    readln;
    halt;
  end;
```

```
sum1 := 0; sum2 := 0;  B1 := 0; B2 := 0; Offset1 := 753;  OffSet2:= 765;
  port6 :=0;

ResetPort6_0(Port6);   Fir := False;  LLL := 0;     GetPort7(P7In);
time:=0;


while true do
begin

 if keypressed=true then
  begin
    writeln(offset1:16:6,Offset2:16:5,b1:12:4,b2:12:4); halt;
  end;

 time := time +0.001;
 GetP2(WW);sign := false;
 if (ww and 128 <>0 ) then sign := true;
 if sign then w := -(WW AND 127)
 ELSE W := ww;         W:=w*2.6;
   Inte(sum1,sum2,Offset1,Offset2);

  Predict(Sum1,Sum2,Theta1,Theta2,P1,P2);

  FirP := Fir;
  Irs(Sum1,Sum2,B1,B2,P1,P2,Fir);

  if ((Fir = True) and (FirP = False)) then
     begin
          SetPort6_7(Port6); {Fire pulse}
          SetPort6_1(Port6); {Wind fire pulse}
    {       clrscr; gotoxy(1,1);
     for iii := 1 to size do
     - begin
        jjj := iii+head-1;
        if JJJ > size then JJJ := JJJ-size;


        writeln(iii:4,jjj:4,(theta1[jjj]-b1):10:4,
                     (theta2[jjj]-b2):10:4,
                      P1:10:4,
                      P2:10:4,
                      wx_store[jjj]:10:0,
                      wz_store[jjj]:10:0);
      end;                        }
```

```
      end;

    if (Fir = True) then LLL := LLL + 1;

    if (LLL =  100) then
      begin
         Fir:=False;
         ResetPort6_7(Port6);
         ResetPort6_1(Port6);
         LLL:= 0;
      end;
;
    GetPort7(P7In);  {32 right, 8 up, 16 down, 64 left}

     begin
      If (P7In and 16) = 0 then Sum1 := Sum1 + 0.05;
      If (P7In and 32) = 0 then Sum1 := Sum1 - 0.05;
      If (P7In and 64) = 0 then Sum2 := Sum2 - 0.15;
      If (P7In and  8) = 0 then Sum2 := Sum2 + 0.15;
     end;

{September 8, 1995}

GetPort7(P7In); {Clock control}
while (P7In and 128) = 128 do begin GetPort7(P7In) end;

end;

end. {main}
```

NO. OF COPIES	ORGANIZATION
2	DEFENSE TECHNICAL INFORMATION CENTER DTIC DDA 8725 JOHN J KINGMAN RD STE 0944 FT BELVOIR VA 22060-6218
1	HQDA DAMO FDQ D SCHMIDT 400 ARMY PENTAGON WASHINGTON DC 20310-0460
1	OSD OUSD(A&T)/ODDDR&E(R) R J TREW THE PENTAGON WASHINGTON DC 20301-7100
1	DPTY CG FOR RDA US ARMY MATERIEL CMD AMCRDA 5001 EISENHOWER AVE ALEXANDRIA VA 22333-0001
1	INST FOR ADVNCD TCHNLGY THE UNIV OF TEXAS AT AUSTIN PO BOX 202797 AUSTIN TX 78720-2797
1	DARPA B KASPAR 3701 N FAIRFAX DR ARLINGTON VA 22203-1714
1	NAVAL SURFACE WARFARE CTR CODE B07 J PENNELLA 17320 DAHLGREN RD BLDG 1470 RM 1101 DAHLGREN VA 22448-5100
1	US MILITARY ACADEMY MATH SCI CTR OF EXCELLENCE DEPT OF MATHEMATICAL SCI MADN MATH THAYER HALL WEST POINT NY 10996-1786

NO. OF COPIES	ORGANIZATION
1	DIRECTOR US ARMY RESEARCH LAB AMSRL D D R SMITH 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB AMSRL DD 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB AMSRL CS AS (RECORDS MGMT) 2800 POWDER MILL RD ADELPHI MD 20783-1145
3	DIRECTOR US ARMY RESEARCH LAB AMSRL CI LL 2800 POWDER MILL RD ADELPHI MD 20783-1145
	ABERDEEN PROVING GROUND
4	DIR USARL AMSRL CI LP (BLDG 305)

NO. OF COPIES	ORGANIZATION
3	CDR USASOC DSCR AOFI RI MSI E BROWN SOST CRDNTR FORT BRAGG NC 28307
2	CDR USASOCOM SOST T W WILLIAMS BLDG 102 MACDILL AFB FL 33621-5316
1	OFFICE OF SPECIAL TECH G SHOCK 1530 RIVERVIEW RD FT WASHINGTON MD 20744
1	CDR US ARMY ARDEC AMSTA CCJ S SMALL PICATINNY ARSENAL NJ 07806-5000
2	CDR US ARMY ARDEC AMSTA DSA SA J UNTERKOFLER M DOWNES PICATINNY ARSENAL NJ 07806-5000
1	T HYNES 1921 N GAFFEY SUITE J SAN PEDRO CA 90731

NO. OF COPIES	ORGANIZATION
	ABERDEEN PROVING GROUND
5	DIR US ARMY ATC STEAC AE CA P MCCALL D GRIFFIN STEAC FC M A ROSE G NIEWENHOUS G BREWER
19	DIR USARL AMSRL WM BA W D AMICO T BROSSEAU R MCGEE F BRANDON G BROWN S WANSACK M HOLLIS J CONDON B DAVIS G KATULKA M KREGEL J MCLAUGHLIN AMSRL WM MB R KASTE L BURTON AMSRL WM BF B HAUG R VON WAHLDE T VONG G HAAS J SPANGLER

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	May 2000	Final

4. TITLE AND SUBTITLE

The Inertial Reticle Technology (IRT) Applied to a Remington 700 Sniper Rifle

5. FUNDING NUMBERS

1L162618AH80

6. AUTHOR(S)

Timothy L. Brosseau, Mark D. Kregel, Bailey T. Haug, and John T. McLaughlin

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

U.S. Army Research Laboratory
ATTN: AMSRL-WM-BA
Aberdeen Proving Ground, MD 21005-5066

8. PERFORMING ORGANIZATION REPORT NUMBER

ARL-TR-2231

9. SPONSORING/MONITORING AGENCY NAMES(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Motion of the muzzle of a shoulder-fired weapon occurs during firing because of many factors, such as breathing, trigger pull, and flinching, and can also be introduced as a result of offhand shooting in less than stable shooting positions. This motion can have adverse effects on the capability of the weapon to hit a target because the shooter is unable to accurately position the muzzle of the weapon onto the target as the projectile exits the barrel. Electronic fire control offers the opportunity to extend the range and accuracy that can be achieved by a sniper or a long-range shooter with the implementation of a dynamic ballistic solution and a precise firing time selection. The Inertial Reticle Technology (IRT) has been developed at the U.S. Army Research Laboratory (ARL) as a novel solution to fire control for such applications. Therefore, to improve the accuracy of a shoulder-fired weapon, ARL applied the IRT to a rifle.

This report presents the complete details of how the IRT was applied to a Remington 700 sniper rifle, along with analysis of long-range live-fire test data fired by a test engineer and an Army sniper.

14. SUBJECT TERMS

Inertial Reticle Technology, Remington 700 sniper rifle, M118 LC ammunition, video riflescope, flat display monitor, quartz rate sensor, firing solenoid

15. NUMBER OF PAGES

39

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18 298-102

INTENTIONALLY LEFT BLANK.

USER EVALUATION SHEET/CHANGE OF ADDRESS

This Laboratory undertakes a continuing effort to improve the quality of the reports it publishes. Your comments/answers to the items/questions below will aid us in our efforts.

1. ARL Report Number/Author ARL-TR-2231 (Brosseau) Date of Report May 2000

2. Date Report Received ______

3. Does this report satisfy a need? (Comment on purpose, related project, or other area of interest for which the report will be used.) ______

4. Specifically, how is the report being used? (Information source, design data, procedure, source of ideas, etc.) ______

5. Has the information in this report led to any quantitative savings as far as man-hours or dollars saved, operating costs avoided, or efficiencies achieved, etc? If so, please elaborate. ______

6. General Comments. What do you think should be changed to improve future reports? (Indicate changes to organization, technical content, format, etc.) ______

CURRENT ADDRESS

Organization

Name E-mail Name

Street or P.O. Box No.

City, State, Zip Code

7. If indicating a Change of Address or Address Correction, please provide the Current or Correct address above and the Old or Incorrect address below.

OLD ADDRESS

Organization

Name

Street or P.O. Box No.

City, State, Zip Code

(Remove this sheet, fold as indicated, tape closed, and mail.)
(DO NOT STAPLE)